(A-69996/RMA)

SYSTEM AND METHOD FOR REDUCING UNAUTHORIZED ACCESS BY PROCEDURAL MESSAGES EXECUTING IN A COMPUTER SYSTEM TO COMPUTER SYSTEM OR MEMORY OR PROGRAMS OR DATA STORED THEREIN

I Claim:

1. A method of maintaining anti-hacking security in a computer system that executes procedural messages using native code to carry out the procedures of the message, said method comprising the steps of:

native code carrying out the procedures of the message allocating, in a single operation, one contiguous memory block range having a single memory boundary position as a buffer for storage;

protecting the allocated storage buffer from overflow by:

reducing the number of operations the native code uses to carry out the procedures of the message that obtain memory pointers to the allocated buffer; and

checking attempts to access a memory locations outside of the allocated single memory block range only against the single memory boundary position of the single buffer memory block range;

so that the likelihood that a computer system hacker can create a buffer overflow and thereby obtain access to other memory ranges to gain entry or control over functions or data of the computer system is reduced.

2. The method in Claim 1, wherein the computer system includes a story player device.

3. The method in Claim 1, wherein computer code to perform memory checking is uniform and compact.

4. The method in Claim 1, wherein a common core of instructions operate on memory.

5. The method in Claim 1, wherein a hacker attempting to produce a memory buffer stack overflow in order to introduce executable code into the system is substantially prevented by the single memory range allocation and checking.

6. The method in Claim 1, wherein the computer system provides more stable operation as a result of the predictable memory operating environment than would be available with conventional memory operating environments.

7. The method in Claim 1, wherein the message procedures include instructions which sub-allocate all memory regions from said single memory block.

8. The method in Claim 1, wherein the message procedures include instructions which can cause said single memory block to be destroyed and reallocated when different parts of the message are

executed, thereby providing procedural flexibility while avoiding the complexities normally associated with memory garbage collection algorithms.

9. The method in Claim 8, wherein the message procedures include at least one instruction which can preserve some or all parts of the data stored in said single memory block in a second allocated memory block, which is itself also checked to make sure accesses outside of the second allocated memory block are never made while said single memory block is being reallocated.

10. The method in Claim 9 where said second allocated memory block is always available during execution of said procedural messages and accesses are checked to be contained within one of the two allocated memory blocks.

11. A computer program product for use in conjunction with a computing machine and including a program module stored on a tangible medium, said program module including instructions for directing operating of the computing device to maintain security in a computer system that executes procedural messages using native code to carry out the procedures of the message, said instructions including instructions for:

native code carrying out the procedures of the message allocating, in a single operation, one contiguous memory block range having a single memory boundary position as a buffer for storage;

protecting the allocated storage buffer from overflow by:

reducing the number of operations the native code uses to carry out the procedures of the message that obtain memory pointers to the allocated buffer; and

checking attempts to access a memory locations outside of the allocated single memory block range only against the single memory boundary position of the single buffer memory block range;

so that the likelihood that a computer system hacker can create a buffer overflow and thereby obtain access to other memory ranges to gain entry or control over functions or data of the computer system is reduced.

12. A system comprising:

means for hardware architecture neutral computer program language, structure and method for execution;

means for autonomous generation of customized file having procedural and data elements from non-procedural flat-file descriptors;

means for intelligently scaling message procedural/data sets to adapt the procedural/data sets to receiver attributes and maintain message intent;

means for an intent preserving message adaptation and conversion system and method for communicating with sensory and/or physically challenged persons;

means for searching and selecting data and control elements in message procedural/data sets for automatic and complete portrayal of message to maintain message intent;

means for adapting content for sensory and physically challenged persons using embedded semantic elements in a procedurally based message file;

means for forward and backward content based version control for automated autonomous playback on client devices having diverse hardware and software;

means for reducing unauthorized access by procedural messages executing in a computer system to computer system or memory or programs or data stored therein;

means for self-directed loading of an input buffer with procedural messages from a stream of sub-files containing sets of logical files;

means for device-neutral procedurally-based content display layout and content playback;

means for thin procedural multi-media player run-time engine having application program level cooperative multi-threading and constrained resource retry with anti-stall features;

means for streaming multimedia-rich interactive experiences over a communications channel; and

means for cooperative application-level multi-thread execution including instruction retry feature upon identifying constrained system resource.